International Journal on Artificial Intelligence Tools

AN AGENT-BASED APPROACH TO INFERENCE PREVENTION IN DISTRIBUTED DATABASE SYSTEMS *

JAMES TRACY, LIWU CHANG, and IRA S. MOSKOWITZ
Center for High Assurance Computer Systems, Mail Code 5540
Naval Research Laboratory
Washington, DC 20375, USA

We propose an *inference prevention agent* as a tool that enables each of the databases in a distributed system to keep track of probabilistic dependencies with other databases and then use that information to help preserve the confidentiality of sensitive data. This is accomplished with minimal sacrifice of the performance and survivability gains that are associated with distributed database systems.

Keywords: confidentiality, database inference, distributed databases, agent, Bayesian networks, information hiding

1. Introduction

For many applications, distributed database systems are generally understood to provide greater performance and survivability than their centralized counterparts (e.g., [8]). The particular applications of concern in this paper contain data of two classes. The first class, is the public data. This is data that all users may see. The second class, is the sensitive data, which is restricted to only certain users. As in a centralized database system, it is often possible for a user to infer sensitive information from publicly available information by exploiting probability dependency relationships. We refer to a compromise of the confidentiality of sensitive data in this way [3,6,15] as "database inference." Distributed databases present challenges to inference prevention methods that are not present in centralized schemes [10,1]. This is because each database in a distributed system does not contain all the data that is necessary to learn the gamut of the possible inference possibilities. Therefore, in order to successfully prevent inference, or to minimize inference, each database must take into account how its data relates to data stored in the other databases in the system.

*This is an invited version of the same titled paper [14] which was presented at ICTAI 2002, Arlington, VA, USA. Research supported by the Office of Naval Research. Contact author: LiWu Chang `lchang@itd.nrl.navy.mil`

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2003		**00-00-2003 to 00-00-2003**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
An Agent-Based Approach to Inference Prevention Distributed Database Systems	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Research Laboratory,Center for High Assurance Computer Systems,4555 Overlook Avenue, SW,Washington,DC,20375**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**16**	

We propose an *inference prevention agent* as a tool that enables each of the databases in a distributed system to keep track of probabilistic dependencies with other databases and then use that information to help preserve the confidentiality of sensitive data. This is accomplished with minimal sacrifice of the performance and survivability gains that are associated with distributed database systems.

2. Background on Distributed Database Inference

2.1. *Database Inference*

Information to be protected in a database may be spread among many attribute values. In this paper, for the sake of simplicity, we consider the case in which sensitive data are associated with only one particular attribute.

Example: (We summarize the example from [1]). In a specific medical database the sensitive information that we are trying to protect from public disclosure, is the report of a patient's AIDS diagnoses. We use the terms *High database* and *Low database* to indicate, respectively, the portions of a database viewed by a database manager (the High user) and a generic (Low) user. A High user has access to both the public and the sensitive data stored in the database, whereas a Low user only has access to the public data.

Table 1. D_H - *sample medical records* (the 1^{st} database)

(H: hepatitis; D: depression; A: AIDS; T: transfusion)

key	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
H	n	y	y	y	n	n	y	y	y	n	n	y	n	y	n	y	n	y	n	y
D	n	y	y	y	y	n	n	n	y	n	y	n	n	y	y	n	y	y	y	n
A	n	n	y	y	n	n	n	y	y	n	y	n	n	y	n	n	n	y	n	n
T	n	n	y	n	n	n	n	y	n	n	y	n	n	n	n	n	n	y	n	y

Our medical database usually consists of attributes that are related to the patient's background (e.g., age, address) and those that are related to the medical diagnosis. We are interested in studying the probabilistic influence of public data upon the sensitive medical diagnosis. (See [2,3] for the treatment of background information.) Table 1 is the medical database for AIDS diagnosis which contains 20 data records (i.e., patients), which are uniquely identified by their key, and four attributes (excluding the key) (i.e., "hepatitis," "depression," "AIDS," "transfusion"). Each attribute has two values, with a 'y' indicating the occurrence of the (diagnosis) result and an 'n' indicating otherwise.

Table 1 shows the High view (denoted here as DH) in our discussion. Note that having one disease (e.g., "AIDS") often causes the occurrence of another physical disorder (e.g., "mental depression"). Consequently, knowing the diagnosis of a physical disorder may lead to the inference of the sensitive information (i.e., "AIDS") about a patient. Thus, protecting information about one disease may require the protection of other probabilistically related records. In this paper, as

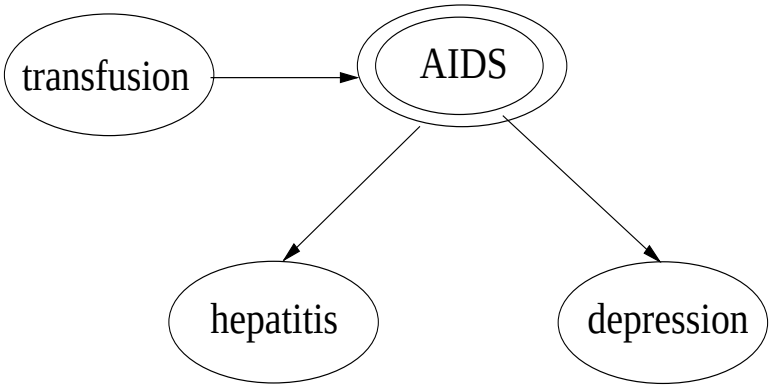

Fig. 1. A Bayesian Network for Sample Medical Records An attribute is denoted by a node. An arrow indicates the probabilistic dependency between two attributes. A double circle denotes that the attribute is sensitive.

in [1], a Bayesian net (network) representation is used to describe the probabilistic relationship. A corresponding Bayesian net representation is given in Figure 1 (see [5,11,12] for details on how to construct a Bayesian net), which shows that "AIDS" may affect the consequence of both "hepatitis" and "mental depression," and also shows that a cause of "AIDS" is a blood transfusion.

Table 2 shows the database that is *considered* for release to the public—the Low database (denoted here as DL). In Table 1 and 2, a patient is identified by its key. The threat we are concerned with is that of Low inferring sensitive relations about the AIDS diagnosis.

In DL the dashes represent data that is considered sensitive and, thus, is not released. Note that Table 2 is only in a *tentative* form for release to the public. One must first *consider* the *inferences* that may be obtained. If these inferences leak sensitive information (who has AIDS), then less information should be considered as publicly releasable. A *target attribute* T is an attribute that has dashes in it (from Low's viewpoint). Thus, T represents sensitive information. We wish to lessen any inference that a Low user may attempt to draw about the *target node*, which is the representation of the target attribute in the Bayesian net (sensitive information). Since data are not completely revealed, the corresponding Bayesian net structure (Figure 2) for DL differs from that of DH. The challenge for a Low user who is attempting to discern sensitive information is to restore the missing information from Table 2. Note that Table 2 still contains the "AIDS" attribute, even though the values are all missing. This is because we take the paranoid view that the Low user knows what the sensitive attribute is.

We continue our paranoia by assuming that the Low user obtains the prior knowledge (say, from previous studies) about the dependency relationship between

Table 2. D_L - *sample medical records as seen by Low user*

(H: hepatitis; D: depression; A: AIDS; T: transfusion)

key	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
H	n	y	y	y	n	n	y	y	y	n	n	y	n	y	n	y	n	y	n	y
D	n	y	y	y	y	n	n	n	y	n	y	n	n	y	y	n	y	y	y	n
A	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
T	n	n	y	n	n	n	n	y	n	n	y	n	n	n	n	n	n	y	n	y

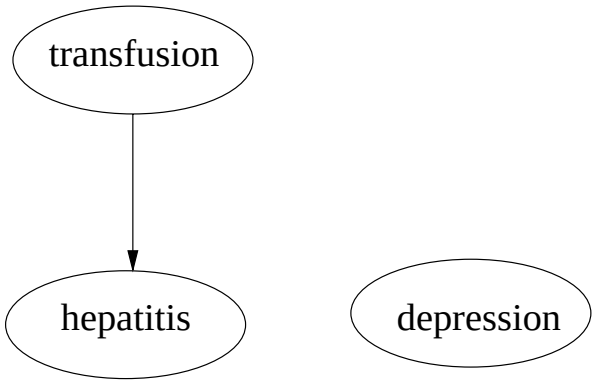

Fig. 2. The Bayesian Network as Constructed by Low User

"AIDS" and the three attributes "mental depression," "hepatitis" and "transfusion." (The dependency relationship is described in Figure 1.) With this dependency knowledge, together with data from the Low database, the Low user may be able to restore the hidden sensitive data. For instance, one may assign a set of values to the hidden attributes that maximizes the sample probability of the entire database [2]. In fact, the technique of probability maximization results in having the restored values be equal to the values of "transfusion." Such a restored Low database is shown in Table 3.

Table 3. *sample medical records as restored by Low user*

(H: hepatitis; D: depression; A: AIDS; T: transfusion)

key	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
H	n	y	y	y	n	n	y	y	y	n	n	y	n	y	n	y	n	y	n	y
D	n	y	y	y	y	n	n	n	y	n	y	n	n	y	y	n	y	y	y	n
A	n	n	y	**N**	n	n	n	y	**N**	n	y	n	n	**N**	n	n	n	y	n	**Y**
T	n	n	y	n	n	n	n	y	n	n	y	n	n	n	n	n	n	y	n	y

Compared with the original values in Table 1, the restored values of Table 3 differ in just 4 places (as shown in the boldface font). This 0.8 chance to make a correct guess is unacceptable. The threat of potential restoration suggests the inadequacy of just hiding the AIDS diagnoses. Therefore, we shall mitigate the

inference by not releasing certain non-sensitive (public) information that *can* lead to probabilistic inferences about the sensitive information [2].

2.2. *Inference in Distributed Databases*

Current information downgrading techniques assume that data come from a single source. However, in the real world there may be several databases [10] in the same context. These databases may have impact upon the sensitive information of the original downgraded database. The inference problem must take into account the impact from different databases. We propose an agent-based tool to help with this inference problem. Note that the multiple databases may have exactly the same structure or overlapping contents. In this paper we assume that each database is willing to hide some of its data from users in order to preserve the confidentiality of not only its own data, but also that of other databases. Such a scheme does not necessarily require each database to know the exact confidentiality requirements of any of the other databases.

Given two databases containing information on the same objects (which can be uniquely identified and joined if the database is in the form of a relational table), the problem that we try to solve is the inference of sensitive data in one database from public data from a different database. The possible interactions for two databases (in the form of relational tables, with schemes $R_1(a_1, a_2, \cdots, a_k)$, and $R_2(b_1, b_2, \cdots, b_l)$) are the following:

(i) R_2 augments R_1 with data records.
(ii) R_2 augments R_1 with different attributes.
(iii) R_2 augments and changes both records and attributes of R_1.

The 1^{st} and 2^{nd} types of interactions respectively correspond to horizontal and vertical combination of databases (in the form of a relational table). What we consider here is when two databases are in different contexts (or, applications), but have overlapped attributes (i.e., the 3^{rd} type of interaction). Also, we assume data records of the two databases come from the same sample population, but attribute values of some objects may be unknown. Data exchange may involve metarules or direct data records.

Here, the form of direct data records will be used in our discussion. Data transferred from the second database may or may not have direct impact on the sensitive information of the first database. The High user will integrate some, but not all, publicly released information from different databases that may cause the disclosure of sensitive data. Combinations of all data may render inference analysis an impractical task due to the huge volumes of data. This is why we propose a Bayesian analysis. We shall analyze the impact based on network dependency properties ([12]), and our practical data sanitization policies, with the following databases.

Consider Table 4, where data shows the diagnosis of Non-Hodgkins lymphomas (NHL) disease. The database manager of the NHL table may observe that a NHL

patient is highly likely to also be an AIDS patient. Thus, data in Table 4 cannot be released if the database manager of the NHL database also agrees to the sanitization/downgrading principle that AIDS data must be secured. Based upon a Bayesian net model of Table 5, data are likely to imply that low thyroid function causes mental depression, which in turn causes high blood pressure. The inference concern is that for a mentally depressed patient, information about low thyroid function would have (negative) impact upon the belief of AIDS diagnosis. (See [9] for details). The degree of impact partially depends on the correlation between "AIDS" and "mental depression," and it can be tested with available data. On the other hand, knowing the state of mental depression would block the impact of blood pressure knowledge. Table 6 is an illegal drug abuse database. Data from Table 6 shows the frequency with which an illegal drug user either takes intravenous injections or smokes. The data indicates a drug injector is likely to have hepatitis. The relationship between "AIDS" and "illegal drug abuse" is not given in Table 6.

Table 4. *NHL cancer database* (the 2^{nd} database)

(N: NHL cancer; A: AIDS)

key	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
N	n	n	y	y	y	n	y	y	n	y	y	y	n	n	y	y	n	y	n
A	n	n	y	y	n	n	y	y	y	y	y	y	n	y	y	y	n	y	n

Table 5. *thyroid database* (the 3^{rd} database)

(D: depression; P: blood pressure; Y: low thyroid)

key	1	2	3	6	7	21	9	22	11	23	13	14	15	24	17	25	19	20
P	n	y	y	y	n	n	n	n	n	y	y	n	y	y	n	n	y	y
D	n	y	y	n	n	y	y	y	y	y	n	y	y	y	y	n	y	n
Y	n	y	n	n	n	y	n	y	n	y	n	y	y	n	y	n	y	n

Table 6. *illegal drug abuse database* (the 4^{th} database)

(I: intravenous injection; S: smoke; H: hepatitis)

key	1	23	2	3	26	6	27	8	10	28	25	12	13	29	30	31	17	18	19	20
I	n	y	n	y	y	n	y	y	n	y	y	n	n	y	y	y	n	y	n	y
S	y	n	y	y	y	y	y	y	y	n	y	y	y	y	y	n	y	y	y	n
H	n	y	y	y	y	n	y	y	n	y	y	y	n	y	y	y	n	y	n	y

(These databases all contain different set of attributes. However, they all have at least one attribute, a key, in common. There is some overlap in the objects they describe. Overlap occurs in this example when objects in multiple databases have the same key. These three databases, in combination with the sample medical records database (Table 1) represent possible components of the type of distributed database system that we are addressing. We focus here on databases that deal with only one class of objects - subjects of a medical study. However, we believe that our approach may be generalized to apply to databases that deal with multiple classes of objects.)

However, for a drug taker, an intravenous injection is unfortunately often a form

Table 7. *modified sample medical records*

(H: hepatitis; D: depression; A: AIDS; T: transfusion)

key	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
H	n	y	?	y	n	n	y	y	y	n	n	y	n	y	n	y	n	y	n	y
D	n	y	y	?	y	n	n	n	y	n	y	n	n	y	y	n	y	y	y	n
A	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
T	n	n	?	n	n	n	n	?	n	n	y	n	n	n	n	n	n	y	n	y

of blood transfusion; one can infer that an illegal drug user is often also an AIDS patient.

2.3. *The Rational Downgrader*

Building Bayesian belief nets and other conceptual models can identify areas in which there is a potential for inference. Additional steps, however, are required, in order to actually prevent inference from occurring. We refer to such measures as "parsimonious downgrading." In other words, we revisit our initial downgrading decisions and adjust them to lessen inference. They generally involve obscuring (by downgrading less than we planned on) certain attributes in records presented to a Low user in order to prevent it from inferring the values of attributes labeled High in the database. There are a number of factors that complicate the downgrading process. One is that the number of attributes that need to be obscured for any given record depends on the values of the record's attributes. Therefore, the actions taken in downgrading are not uniform across all of a Low user's query possibilities. Another factor that complicates downgrading is that an overly aggressive downgrading strategy can render the responses to the Low user's queries useless. Moskowitz and Chang [7] developed (prototype version) the Rational Downgrader in order to address these difficulties. The Rational Downgrader is model of a system for downgrading a database in an "intelligent" manner. As discussed in [2], the Rational Downgrader incorporates metrics that enable it to quantify both the reduction in usefulness and the level of inference with respect to High data that results from the obscuring of various attributes in the records available to Low. Using these metrics it is able to conduct a directed search of attribute values to be hidden (referred to here as downgrading strategies) and select one that maximizes usefulness while at the same time minimizes the possibility of inferring High data. The outputs of the Rational Downgrader are records that have been modified in order to obscure certain attributes, as in the Table 7.

As discussed before, the dashes represent information that we do not want to release to Low. The '?'s represents additional data that is not released to low in order to lessen inference.

3. The Agent-Based Approach to Implementation

3.1. *Overall Architecture*

The introduction of the Rational Downgrader into a distributed database application raises several implementation issues. The first issue is that the computation of a downgrading strategy performed by the Rational Downgrader is very complex. Performing this computation for every query to the database would cause an unacceptable increase in the database application's response time. Another issue is that the Rational Downgrader could represent a single point of failure and communication bottleneck that would undermine the main advantages of distribution. A third implementation issue is that distributed database applications often involve heterogeneous database management systems. Incorporating the functionality of the Rational Downgrader into each database management system would greatly complicate the development of the Rational Downgrader and the maintenance of the database management systems.

In order to address these issues, we have developed a set of requirements for the architecture of the Rational Downgrader mechanism.

(i) The architecture must allow downgrading strategies to be computed infrequently and reused.
(ii) The architecture must enable the execution of downgrading strategies to be distributed in the same manner as the storage of data.
(iii) The architecture must be cleanly separated from each of the database management systems and be based on a standard communication protocol.

An agent-based architecture meets these requirements (for a similar use of agents see [13] and [4]). In this context, we use the term "agent-based" to describe a scheme in which the execution of a downgrading strategy is distinguished from its creation and is delegated to a number of independent processes. Such an architecture allows downgrading strategies to be encapsulated in the form of inference prevention agents. Because the agents are then capable of carrying out the strategies themselves, for as long as they are valid, they eliminate the need for recomputing downgrading strategies upon every access request. In addition, the autonomous nature of the agents allows them to be deployed in an environment that is separate from the one in which they were created.

The agents can be deployed on the same machines as the databases that they filter, so that a single point of failure and communication bottleneck is avoided. Finally, because the agents have an implementation that is completely unaware of the details of any database management system, the difficulty of development and maintenance would not drastically increase when they were applied to a heterogeneous distributed database application. Figure 3 is a visual representation of the proposed architecture.

3.2. *Design of the Inference Prevention Agent*

The inference prevention agent will be a production system [16] that is associated with a particular database in the distributed database application. The facts in the

agent's production system are the records in the database. The rules are generated by the Rule Generator, which will be described in detail later. The rules check for certain combinations of values in attributes and specify attributes that should be hidden. An example of a possible rule is provided below.[1]

RULE 1: IF H = 'y' AND T = 'y' THEN HIDE H

Suppose that we have an agent that contains RULE 1, and a Low user specifies the query:

SELECT H,T FROM TABLE_1 WHERE KEY = 3

The agent first must retrieve (see Figure 4) the facts it needs from the database management system in order to apply its rules. Rules may apply to any of the attributes in the database, so the agent must expand the user's query to include all attributes. The agent would then make the following SQL query to the local database management engine:

SELECT * FROM TABLE_1 WHERE KEY = 3

The database management engine's response to this query consists of the following record as shown in Table 8. The database management system's response provides the agent with the facts its needs for its execution. The agent will detect that the facts in this case match RULE 1 as specified above.

Accordingly, it will substitute '?' for 'y' in the attribute that specifies whether patient 3 has hepatitis or not. Note that the downgrading was accomplished without any communication with any other databases, without any communication with the Rule Generator, and without any modification to the database management systems.

3.3. *Agent Communication*

Cases will arise when facts from the local database alone cannot be used to evaluate rules. Suppose that, for example, the rule in question were as follows:

RULE 2: IF H = 'y' AND T = 'y' AND Y = 'y' THEN HIDE H AND I

Suppose also that Y (thyroid) data resided on a separate machine from the hepatitis information. In order to handle such cases, agents will need to communicate with remote databases. The nature of the agents' communication may be illustrated with an example. Let us say that Agent A receives the query from the Low user:

SELECT H FROM TABLE_1 WHERE H = 'y'

[1] We deal only with SELECT SQL statements in the this paper, as they are the only type that affect our ability to preserve confidentiality.

Clearly there is the possibility that RULE 2 may be activated. As in the first example of rule evaluation, Agent A will make a broader query from its local database.

SELECT * FROM TABLE_1 WHERE H = 'y'

which produces the result shown in Table 9. This query will provide Agent A with the hepatitis information and transfusion information, but not the depression information that it needs to evaluate the rule. Thus Agent A needs to locate the database that can provide access to the depression information. As part of the agent design, each agent will be given an attribute directory that specifies which databases contain which attributes. The contents of this directory will be specified at rule generation time.

Accordingly, Agent A consults this directory and discovers that the database responsible for Y (thyroid) is TABLE_5. Now Agent A needs to collect the thyroid attribute for all the records it shares with Table 5 that have 'y' in the Y attribute, 'y' in the H attribute, and 'y' in the T attribute. Unfortunately, while TABLE_5 may know the set of records for which Y='y', it does not know the set for which the other conditions hold, because it does not contain the hepatitis or transfusion information. It should not send the entire set of records for which Y = 'y' because this may be prohibitively large. Agent A needs to specify a subset of records to which TABLE_5 must apply its query of the thyroid attribute. In this case, that subset is all the records in Table 1 for which H='y' and T='y'. Agent A can specify these records using the key that the local database and TABLE_5 shares. The SQL for such an operation in this example would be:

SELECT Y FROM TABLE_5 WHERE Y = 'y' AND (KEY = 3 OR KEY = 8 OR KEY = 18 OR KEY = 20)

The result of this query is exactly the set of records to which Rule 2 applies.

Table 8. result of SELECT * FROM TABLE_1 WHERE KEY =3

(H: hepatitis; D: depression; A: AIDS; T: transfusion)

key	3
H	y
D	y
A	y
T	y

Table 9. result of SELECT * FROM TABLE_1 WHERE H ='y'

(H: hepatitis; D: depression; A: AIDS; T: transfusion)

key	2	3	4	7	8	9	12	14	16	18	20
H	y	y	y	y	y	y	y	y	y	y	y
D	y	y	y	n	n	y	n	y	n	y	n
A	n	y	y	n	y	y	n	y	n	y	n
T	n	y	n	n	y	n	n	n	n	y	y

Table 10. combined High database

(H: hepatitis; D: depression; A: AIDS; T: transfusion; I: drug injector; S: smoke; P: blood pressure; Y: low thyroid N: non-Hodgkins Lymphomas).

key	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
H	n	y	y	y	n	n	y	y	y	n	n	y	n	y	n	y
D	n	y	y	y	y	n	n	n	y	n	y	n	n	y	y	n
A	n	n	y	y	n	n	n	y	y	n	y	n	n	y	n	n
T	n	n	y	n	n	n	n	y	n	n	y	n	n	n	n	n
I	n	n	y	*	*	n	*	y	*	n	y	n	n	*	*	*
S	y	y	y	*	*	y	*	y	*	y	n	y	y	*	*	*
P	n	y	y	*	*	y	n	*	n	*	n	*	y	n	y	*
Y	n	y	n	*	*	n	n	*	n	*	n	*	n	y	y	*
N	n	n	y	y	n	y	n	y	y	n	y	n	n	y	n	n

17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
n	y	n	y	*	*	y	*	y	y	y	y	y	y	y
y	y	y	n	y	y	y	y	n	*	*	*	*	*	*
n	y	n	n	*	*	*	*	*	*	*	*	*	*	*
n	y	n	y	*	*	*	*	*	*	*	*	*	*	*
n	y	n	y	*	*	y	*	y	y	y	y	y	y	y
y	y	y	n	*	*	n	*	y	y	y	n	y	y	n
n	*	y	y	n	y	y	y	n	*	*	*	*	*	*
y	*	y	n	y	n	y	n	n	*	*	*	*	*	*
n	y	n	*	*	*	*	*	*	*	*	*	*	*	*

3.4. *Design of the Rule Generator*

The Rule Generator executes far more infrequently than the agents. Its purpose is to create new agents from time to time so that the inference prevention strategy may closely reflect the probability dependency relationships among the databases in the system. In order to perform its task, it needs a comprehensive view of the entire distributed database system. Such a view may be constructed by performing an outer join on the key that the databases share, as shown in Table 10.

Table 10 shows the combination of the original High data with records of illegal drug injection takers and records of the thyroid function, where the '*' denotes attribute values that are unknown because data records of these databases are not taken from a completely overlapped population. (Here, the assumption is that the database manager of the "AIDS" database is able to identify and select patients from the other two databases based on a common key.) The dependency relationship between attributes of the combined database is given by the Bayesian net of Figure 5. Note that Figure 5 has resulted from composing dependency relationships derived from these three databases, together with the knowledge about the relationship between intravenous injection and blood transfusion, and is not generated from combined data. It is known that the generation of a reliable complex network model in general demands large volumes of data. Here, we assume that the dependency relationship derived from each individual database is preserved in the combined database. For our current example, this assumption (referred to as dependency inheritance under combination) seems to be valid. It is useful in handling the combination of multiple large databases, yet its validity has not been formally

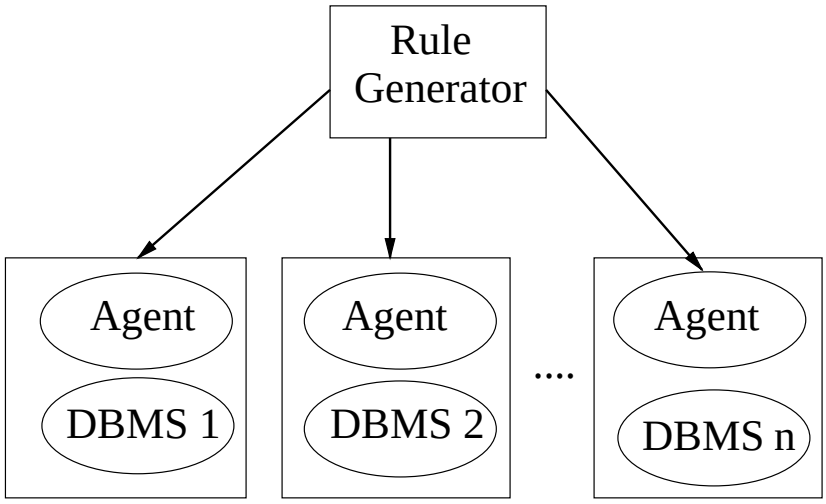

Fig. 3. Architecture of Agent-based Rational Downgrader

proven. We shall investigate this issue in the future work.

The outer join of the databases is used to train the Bayesian net. Note that the rule generator does not retain the outer join it creates after the training is over. Nor does it interact with the users. It is not meant to be an operational substitute for the databases themselves. As soon as the new agents are created, the resources devoted to storing the outer join it created may be reclaimed for other purposes.

The outer join need not be performed on all the instances in the distributed database at the same time. We can train a Bayesian networks on subsets of the instances, as long as all of the existing attributes of the records in those subsets are present. We can then combine the results of these training sessions into a single Bayesian network by properly weighting the conditional probabilities in each network by the number of records used to train the network and calculating a weighted average (referred to here as horizontal combination). Horizontal combination makes the storage space requirements for the rule generator reasonable.

The rules are derived from the trained Bayesian net by analyzing the influence of an attribute on the sensitive target attribute. There are many possible approaches to deriving filtering rules from a Bayesian net. Our approach has been to use conditional probability as a measure of the influence of an attribute, A, on a sensitive target attribute, T. Where v and u denote values of attributes and Bn stands for the Bayesian net model in Figure 5, the probability that T = v given A=u is written as $Prob(T = v|A = u, Bn)$. We treat this value as a measure of the potential of inferring that A=u when T=v, given the Bayesian net structure Bn. In the present medical example, the two attribute values that have the strongest influence on the positive diagnosis "AIDS" (A = 'y') are "NHL cancer" positive (N=y) and "blood transfusion" positive (T=y), where the measures are 0.88 and 0.8, respectively. In

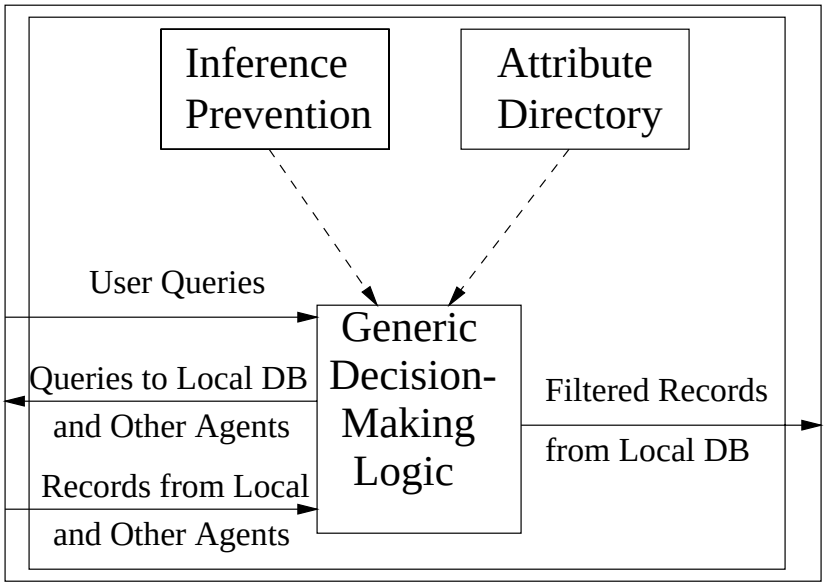

Fig. 4. The Design of An Individual Agent

fact, attribute values that have greater influence usually arise from those attributes that are directly probabilistically related to "AIDS." In Figure 5, in descending order by degree of influence, they are T=y, I=y, H=y and D=y. (Note that we exclude "NHL cancer" from our rule generation example because it has so a strong dependency relationship with the "AIDS" attribute that it obviously requires hiding at all times). Generally speaking, rules will include those attribute values with high influence measures relative to the rest of the attribute values. If we assume for the sake of simplicity that we consider only the four highest ranking attribute values, the Rule Generator will exhaustively evaluate the effect of hiding some of the four attribute values against the belief measure of "AIDS" being positive. It then selects the combinations that cause the change of the belief of "AIDS" being positive beyond a given threshold. For example, suppose the attribute values of interest are the four with highest influence measures, i.e., T=y, I=y, H=y and D=y. Let v_1, v_2, v_3 and v_4 denote the original values of T, I, H and D, respectively, and v_1', v_2', v_3' and v_4' are the modified values where a v_i' can be either v_i or '?'.[2] In Figure 5, for a given inference prevention threshold τ, we compute

[2]We do not consider other values of the i^{th} attribute other than its original value, v_i, and the non-informative value '?' in this paper, because it is our policy that we do not introduce erroneous data for the obvious pragmatic reasons.

$$\alpha = |Prob(AIDS = y|T = v_1, I = v_2, H = v_3, D = v_4, Bn) -$$
$$Prob(AIDS = y|T = v_1', I = v_2', H = v_3', D = v_4', Bn)|$$

and record those modifications where $\alpha > \tau$. We then generate a set of inference prevention rules based on these modifications. In the current example, one such rule could be:

IF T='y' AND I='y' AND H = 'y' AND D = 'y' THEN HIDE T AND I

The search for the inference prevention rule sets is biased toward attribute values with higher influence measures. It is clear that the number of inference prevention rules is related to the value of the inference prevention threshold τ. Depending upon the level of security required in an application, τ can be lowered to ensure those security requirements. The reduction in inference that results from hiding T and I is significant enough that it is not necessary to also hide H and D. The set of rules with which the decision of an agent is made is a summary of the probabilistic information embedded in a probabilistic network. The decision of the size of the set rules depends upon the precision required by applications. In future work we would like to analyze the effect of the size of the inference prevention rule set on the achievable inference prevention threshold τ. The construction of a satisfactory Bayesian net in the medical domain is a knowledge intensive effort; its network model is relatively less likely to change. Thus, the set of rules will not vary over short periods of time.

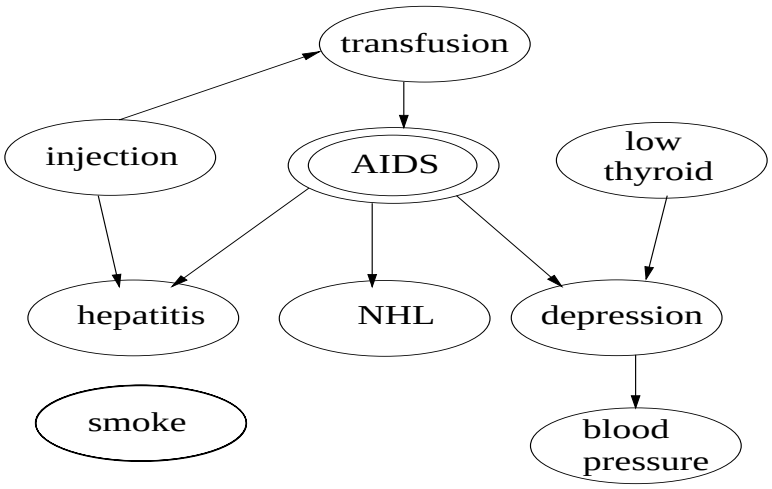

Fig. 5. The Combined Bayesian Network

In this example, the inference prevention rule is deterministic and hence, every arriving datum (or, a record) that satisfies the conditioning part of the rule is

modified. However, as shown in [2], not all data that have the same attribute values are modified in the case of a centralized single database modification - modification stops when a threshold is achieved. Suppose we do not make dynamic modifications to arriving data, but postpone modification for a period of time and then modify data collected over that period based on a centralized method (e.g., [2]). Would we make the same amount of modifications? We do not think it is likely. By using the deterministic inference prevention rules we may over-modify data. To reduce the amount of modifications, we are also investigating *probabilistic* inference prevention rules in which the probability of a rule is given by the frequency of occurrences of its conditioning part in the complete database. Probabilistic rules may give better database performance.

It is worth pointing out that this scheme not only increases the security of confidential data, but also promotes compartmentalization within the distributed database system. While the rule generator has access to High data in all of the databases, it never disseminates the data of one database to any other database in the system. The only information transferred are rules that apply to Low data and the locations of Low attributes in the system. None of this information makes any reference to High attributes. Although our paranoid worst case assumes otherwise, under usual circumstances it would be difficult for a given database administrator to find out the number, names, or probabilistic relationships of attributes in other databases.

4. Conclusion and Future Work

The conclusion of our analysis is that an agent-based tool is well suited to the problem of providing inference prevention capability in a distributed database system. Our rationale for favoring the agent-based approach is summarized by the following list of advantages:

(i) Since the agents work in parallel and are local to the databases, the performance benefit of distribution is not lost. There is no bottleneck through which all queries must pass.
(ii) Similarly, the survivability benefit of distribution is not lost. The potential single point of failure represented by a centralized Rational Downgrader is avoided.
(iii) The compartmentalization provided by a distributed scheme is preserved. Databases can prevent the inference of sensitive data in other databases without knowing exactly what the nature of that data is.
(iv) Interoperability is insured. Heterogeneous databases can participate in the inference prevention if they are compliant with the SQL standard.
(v) A separation of concerns is maintained. Changes to the inference prevention scheme do not require changes to the database management systems.

Some additional work may be required before our approach may be applied to operational systems. In particular, we would like to verify the hypothesis of dependency inheritance of Bayesian belief nets under combination, describe formally how our approach may be generalized to databases that describe more than one class

of objects, and provide a more formal description of our algorithm for generating inference prevention rules from Bayesian belief nets.

Acknowledgments

We thank the anonymous reviewers and Judy Froscher for their helpful comments.

References

[1] Chang, L. and Moskowitz, I. S. (2002) "A Study of Inference Problem in Distributed Database Systems," Data And Applications Security, (eds. Shenoi and Gudes), Kluwer, 2003, (Proc. IFIP WG 11.3 2002).

[2] Chang, L. and Moskowitz, I. S. (2000) "An Integrated Framework for Database Inference and Privacy Protection," Data And Applications Security, (eds. Thuraisingham, van de Riet, Dittrich, and Tari), Kluwer, 2001, pp. 161-172, (Proc. IFIP WG 11.3 2000).

[3] Denning, D. (1980) "Secure Statistical Database with Random Sample Queries," ACM Trans. on Database Systems, 5(3), pp. 291-315.

[4] Gray, R.S., Cybenko, G., Kotz, D., Peterson, R.A. and Rus, D. (2001) "D'Agents: Applications and Performance of a Mobile-Agent System." Software– Practice and Experience.

[5] Heckerman, D. (1996) "Bayesian Networks for Knowledge Discovery," Advances in Knowledge Discovery and Data Mining, AAAI Press/MIT Press, pp. 273-305.

[6] Hinke, T., Delugach, H. and Wolf, R. (1997) "Protecting Databases from Inference Attacks," Computers and Security, Vol. 16, No. 8, pp. 687-708.

[7] Moskowitz, I.S., and L. Chang. (1999) "The Rational Downgrader," Proc. PADD'99, pp. 159-165, London, UK.

[8] Ozsu, M. T., and Valduriez P. (1999) Principles of Distributed Database Systems, Prentice Hall

[9] Pearl, J. (2000) Causality, Cambridge.

[10] Prodromidis, A., Chan, P. and Stolfo, S. (2000) "Meta-learning in distributed data mining systems: Issues and approaches," Advances in Distributed and Parallel Knowledge Discovery, (eds.) Kargupta, H. and Chan, P., AAAI/MIT.

[11] Spiegelhalter, D. and Lauritzen, S. (1990) "Sequential updating of conditional probabilities on directed graphical structures," Networks, 20, pp. 579-605.

[12] Spirtes, P., Glymour, C. and Scheines, R. (1993) Causation, Prediction, and Search. Springer-Verlag, NY.

[13] Tripathi, A., Ahmed, T., Pathak, S., Pathak, A., Carney, M., Koka, M., and Dokas, P. (2002) "Active Monitoring of Network Systems using Mobile Agents,"Proceedings of Networks 2002.

[14] Tracy, J., Chang, L. and Moskowitz, I. S. (2002) "An-Agent-Based Approach to Inference Prevention in Distributed Database Systems," Proc. ICTAI 2002, Washington, DC. pp 413-422.

[15] Thuraisingham, B. (1998) Data Mining: Technologies, Tools and Trends, CRC Press.

[16] Widom, J., and Ceri, S., (1996) Active Database Systems: Triggers and Rules For Advanced Database Processing, Morgan Kaufmann.